Adaptive pattern recognition system for scene segmentation

Toshiro Kubota
Terry Huntsberger, MEMBER SPIE
University of South Carolina
Department of Computer Science
Intelligent Systems Laboratory
Columbia, South Carolina 29208
E-mail: kubota@cs.sc.edu

Abstract. Robust pattern recognition within the Bayesian framework for scene segmentation/boundary detection is often hampered by the presence of textures within natural images. To improve segmentation/boundary detection on natural images, it is necessary to combine multiple features effectively. Two algorithms for combining both color and texture features to assist boundary detection processes are introduced. One combines features through the surface processes and the other through the line processes. The algorithms can be generalized for combining any number of feature sets. [S0091-3286(98)00903-9]

Subject terms: boundary detection; Bayesian model; texture; features; minimization.

Paper ART-121 received June 2, 1997; revised manuscript received Aug. 11, 1997; accepted for publication Aug. 11, 1997.

1 Introduction

In this paper, color features are defined as features obtained by pixelwise operations such as scaling and mean subtraction on the original data, while texture features are defined as features obtained by combination of pixelwise operations and local operations such as averaging and linear filters. The color segmentation implies the segmentation using only color features, while the texture segmentation implies segmentation using only texture features.

Many existing segmentation/boundary detection algorithms deal only with either textured images or nontextured images.[1–3] When texture segmentation algorithms are applied to nontextured images, they often fail to detect the location of boundaries accurately, whereas when the color segmentation algorithms are applied to such images, they can easily obtain accurate boundaries. On the other hand, when the color segmentation algorithms are applied to textured images, they produce spurious boundaries within textured regions and often fail to delineate the region boundaries clearly. Thus, they are not sufficient for natural images, which contains both textured regions and nontextured regions with clear boundaries.

Modeling the image formation process through a discretized system of partial differential equations (PDEs) is one way to address the segmentation/boundary detection problems.[4,5] A model that has gained much attention recently is called the weak membrane model (WMM) and consists of two Markov random fields: one models homogeneous surfaces and is called the surface process and the other specifies discontinuities explicitly and is called the line process.[6–8] With this method, the objective becomes finding the global minimum of the energy function described as

$$E=\int \alpha\|s-g\|^2+\mu\|\nabla s\|^2(1-l)^2+\nu l^2 \, \mathrm{d}x \, \mathrm{d}y, \qquad (1)$$

where $g(x,y)$ is the extracted local features, called the observation; $s(x,y)$ is the surface process and represents a smooth surface that is constrained by the observation; $l(x,y)$ is the line process, which takes either 0 or 1; ∇ is the gradient operator, and $\|\cdot\|$ is the norm in the feature vector space. The Euclidean norm is used in all of our experiments. The second term is called the stabilizer and the third term is the penalty. Throughout the rest of the paper, the integral over the spatial domain is omitted for energy representation, implying that the total energy is the sum of local energy, which can be determined locally by two Markov random fields.

Through minimization of the energy function, the surface process maintains closeness to the observation, and develops into a smooth surface within a boundary where $l=0$. The surface process is allowed to be discontinuous at boundaries where $l=1$. The penalty term penalize having $l=1$ so that it prevents l from being 1 everywhere. As ν increases, the number of boundary pixels in the result decreases.

The energy landscape of Eq. (1) is not convex, therefore, we must use a global optimization technique such as simulated annealing and Hopfield network to obtain a near-optimal solution.[3,6,9]

The left image in Fig. 1 shows a synthetic image with both textured and nontextured regions. When the boundary detection process using the WMM with a color feature is applied to this image, the middle image in Fig. 1 is obtained as the result. The process detected the sharp boundary between the two nontextured regions and also the boundary between the top-textured region (grass texture) and the nontextured region. However, it failed to complete the boundaries between textured regions and between the bottom two regions. It also produced many spurious boundaries within the textured regions.

Now the same membrane model is applied to a texture feature set using Law's texture filters (see Sec. 5 for de-

Fig. 1 Synthetic test image and results of conventional boundary detection algorithms: left, the original synthetic image; middle, a typical result of minimizing the WMM with color feature ($\alpha=0.1$, $\mu=1.0$, and $\nu 0=0.03$); and right, a typical result with texture features ($\alpha=0.1$, $\mu=1.0$, and $\nu 0=0.4$).

tails). The result is the right image in Fig. 1. Note that all boundaries are detected fairly well. However, spurious double boundaries are produced between two nontextured regions, and the boundaries are relatively thicker than the result of the color feature WMM. Thus, the accuracy of the boundary location is not precise.

To achieve robust segmentation/boundary detection on natural images, it is necessary to combine multiple features effectively. This paper proposes algorithms that use both texture and color features and complement the deficiencies observed in the preceding examples. Here, we call the process of combining two sets of features blending. With the WMM, the blending can be done through the region processes s or through the line processes l.

Another important issue related to many segmentation/boundary detection methods is parameter estimation. We have developed a heuristic parameter adaptation scheme to enhance the performance of the Bayesian based boundary detection algorithm.[10] This adaptation technique was modified and applied in conjunction with the blending methods.

Section 2 describes the algorithm for blending through surface processes. Section 3 describes the algorithm for blending through line processes. Section 4 describes a parameter adaptation method to enhance the boundary detection technique. Section 5 provides some experimental results, and Sec. 6 provides a summary and conclusions.

2 Blending through Region Processes

First, the algorithm is described in a general case where M sets of features must be blended for boundary detection. Then it is described in detail for color and texture feature blending.

The main idea is to combine multiple features by normalized weights and to minimize the energy function with respect to the weights to obtain near-optimal weights. The weights are denoted λ_i and satisfy the following constraints:

$$\sum_i \lambda_i = 1, \quad \lambda_i \geqslant 0. \tag{2}$$

Then the weighted features and surface processes are

$$\overline{g_i} = \lambda_i g_i, \tag{3}$$

$$\overline{s_i} = \lambda_i s_i. \tag{4}$$

The total energy of the model is

$$E = \sum_i \alpha_i \|\overline{s_i} - \overline{g_i}\|^2 + \mu \|\nabla \overline{s_i}\|^2 (1-l)^2 + \nu l^2. \tag{5}$$

If we assume

$$\frac{\delta \lambda_i}{\delta x}(1-l)^2 = \frac{\delta \lambda_i}{\delta y}(1-l)^2 = 0, \tag{6}$$

then

$$E = \sum_i [\lambda_i^2 \alpha_i \|s_i - g_i\|^2 + \lambda_i^2 \mu \|\nabla s_i\|^2 (1-l)^2] + \nu l^2. \tag{7}$$

To ensure the assumption of Eq. (6) to be valid, an additional term $\mu \|\nabla \lambda_i\|^2 (1-l)^2$ is added to the energy function. Thus, our final energy function for the feature blending is

$$E = \sum_i [\lambda_i^2 \alpha_i \|s_i - g_i\|^2 + \lambda_i^2 \mu \|\nabla s_i\|^2 (1-l)^2 + \mu \|\nabla \lambda_i\|^2 (1-l)^2] + \nu l^2. \tag{8}$$

Initially, λ_i are all set to $1/M$. Minimization of Eq. (5) is performed not only by updating s_i and l but also by updating λ_i. Solving the Euler-Lagrange equation gives the update rule for s_i as

$$\frac{ds_i}{dt} = -\alpha_i \|s_i - g_i\| + \nabla [\mu (1-l)^2 \nabla s_i]. \tag{9}$$

The update rule for l using the Hopfield network is[9,11]

$$\frac{dl_m}{dt} = \mu \sum_i \|\nabla s_i\|^2 (1-l) - \nu l - A(t) l_m, \tag{10}$$

$$l=\frac{1}{1+\exp\,(-l_m/T)}, \tag{11}$$

where

$$\lim_{t\to\infty} A=0 \quad \lim_{t\to\infty} T=0, \tag{12}$$

and l_m is the internal state variable for an analog neuron in the Hopfield network.[9] The time variable T and A correspond to the temperature and the gain control for the analog neuron, respectively.[9] They start from relatively large values so that the energy landscape is convex with respect to l_m. As they decrease, the energy approaches Eq. (5), and the line process becomes either 1 or 0. A typical scheduling for T and A is

$$T(n)=\eta T(n-1), \quad A(n)=\eta A(n-1), \quad 0<\eta<1, \tag{13}$$

where n is the time index, which is incremented by 1 when the system reaches a stable point.

To satisfy the hard constraints on λ_i in Eq. (2), we assume that increasing λ_i by δ implies that decreasing other $M-1$ λ's by $\delta/(M-1)$. Then the update rule for λ_i is

$$\begin{aligned}\frac{d\lambda_i}{dt}=&-\lambda_i\alpha_i\|s_i-g_i\|-\lambda_i\mu\|\nabla s_i\|^2(1-l)^2\\&+\frac{\lambda_i}{M-1}\sum_{j\neq i}[\alpha_j\|s_j-g_j\|+\mu\|\nabla s_j\|^2(1-l)^2]\\&+\mu\|\nabla\lambda_i\|^2(1-l)^2.\end{aligned} \tag{14}$$

Now consider for blending of color and texture features. In this case, $M=2$. Throughout the paper, processes and parameters related to the color features are denoted by a subscript c and those related to the texture features are denoted by a subscript t. Thus, s_c denotes the color surface processes and s_t denotes the texture surface processes.

Now the energy function of Eq. (8) becomes

$$\begin{aligned}E=&\lambda^2\alpha_c\|s_c-g_c\|^2+\lambda^2\mu\|\nabla s_c\|^2(1-l)^2+(1-\lambda)^2\alpha_t\|s_t\\&-g_t\|^2+(1-\lambda)^2\mu\|\nabla s_t\|^2(1-t)^2+\mu\|\nabla\lambda\|^2(1-l)^2\\&+\nu l^2.\end{aligned} \tag{15}$$

This yields the update rule for λ as

$$\begin{aligned}\frac{d\lambda}{dt}=&-\lambda(\alpha_c\|s_c-g_c\|-\alpha_t\|s_t-g_t\|)-\lambda\mu(\|\nabla s_c\|^2-\|\nabla s_t\|^2)\\&+\nabla[\mu(1-l)^2\nabla\lambda],\end{aligned} \tag{16}$$

or

$$\frac{d\lambda}{dt}=-E_c(s_c)+E_t(s_t)+\nabla[\mu(1-l)^2\nabla\lambda], \tag{17}$$

where E_c and E_t are the sums of the first two terms of the membrane energy [Eq. (1)] for the color and the texture features, respectively. Thus, λ adapts itself such that it weighs features with less energy [in terms of the membrane equation, Eq. (1)] more than the other.

Equation (16) does not guarantee that λ remains in the constrained range [0,1] and λ must be forced to stay within the range if it tends to go out of the range. By scaling two sets of features in similar ranges, λ tends to stay within the range. In our experiment, each feature image is normalized to zero mean and variance 1 prior to the blending process.

3 Blending through Line Processes

With this method, each feature set has its own line process, however, the line processes are constrained to be identical by the term $(l_i-l_j)^2$ in the energy equation. The energy to be minimized is

$$\begin{aligned}E=\sum_i\Bigg[&\alpha_i\|s_i-g_i\|^2+\mu_i\|\nabla s_i\|^2(1-l_i)^2+\nu_i l^2\\&+\frac{\xi}{2}\sum_j(l_i-l_j)^2\Bigg].\end{aligned} \tag{18}$$

The update rule for the surface process is

$$\frac{ds_i}{dt}=-\alpha_i\|s_i-g_i\|^2+\nabla[\mu_i(1-l_i)^2\nabla s_i], \tag{19}$$

and the update rule for the line process is

$$\begin{aligned}\frac{dl_{mi}}{dt}=&\mu_i\|\nabla s_i\|^2(1-l_i)-\nu_i l_i-B(t)\xi\sum_j(l_i-l_j)\\&-A(t)l_{mi},\end{aligned} \tag{20}$$

$$l=\frac{1}{1+\exp\,(-l_{mi}/T)}, \tag{21}$$

where

$$\lim_{t\to\infty} A=0 \quad \lim_{t\to\infty} B=1.0 \quad \lim_{t\to\infty} T=0. \tag{22}$$

The time variable A is the gain control of the Hopfield analog neuron and T is the temperature, as described in the previous section. The variable B begins with a large value and decreases by a rate similar to that of the annealing temperature T decrease. A typical scheduling for B is

$$B(n)=\eta B(n-1)+1, \quad 0<\eta<1, \tag{23}$$

so that it does not fall below 1. It is used to enhance the contribution of the constraint $l_i=l_j$ at early stages of the minimization process when the difference of two line processes are very small [see Eq. (21)]. Without this enhancement, the last term in Eq. (6) is negligible at small T, and it does not contribute to the line process update until the line process establishes its state significantly. At this point, it is not easy to change the state of the line process from the on state to the off state or vice versa.

For the case with color and texture features, the energy function becomes

$$E=\alpha_c\|s_c-g_c\|^2+\alpha_t\|s_t-g_t\|^2+\mu_c\|\nabla s_c\|^2(1-l_c)^2 + \mu_t\|\nabla s_t\|^2(1-l_t)^2+\nu_c l^2+\nu_t l^2+\xi(l_c-l_t)^2. \quad (24)$$

The update rule for the color line process is

$$\frac{dl_{mc}}{dt}=\mu_c\|\nabla s_c\|^2(1-l_c)-\nu_c l_c-B(t)\xi\sum_j (l_c-l_t) - A(t)l_{mc}, \quad (25)$$

followed by Eq. (21). The subscripts c and t must be swapped for the texture line process update rule.

4 Parameter Adaptation

The conventional WMM, which assumes the same energy function throughout the image often produces unsatisfactory results especially when high-contrast regions and low-contrast regions coexist within an image. In order to circumvent this problem, the parameter ν can be adjusted locally based on the local feature statistics. This section briefly describes an effective parameter adaptation technique (for more detail, see Ref. 10).

By employing the mean field approximation on the WMM, the line process can be controlled by[12]

$$l=\frac{1}{1+\exp\,(\nu-\mu|\nabla f|^2/T)}. \quad (26)$$

It is easy to see in Eq. (26) that $(\nu/\mu)^{1/2}$ acts as a threshold for the line process. When the feature gradient is larger than $(\nu/\mu)^{1/2}$ the line process at the location develops into the on state, otherwise develops into the off state.

Our objective through adaptation of the parameter ν is to extract salient boundaries/edges regardless of their illumination intensities. Initially, ν is estimated by

$$\nu=\nu_0\mathcal{G}(x,y)^2, \quad (27)$$

where

$$\mathcal{G}=|\nabla f|*G, \quad (28)$$

and G is a concentric Gaussian filter. Note that $\mathcal{G}$ is squared so that ν matches with $|\nabla f|^2$ in Eq. (26). With this method, ν adapts locally by the local feature gradient density and the global constant parameter ν_0.

Like the surface process, $\nu(x,y)$ is also forced to be smooth within a boundary, and discontinuities are allowed only at the boundary. To ensure this constraint, the same nonlinear diffusion operation that is a part of the surface process update in Eq. (9) is applied at the same rate with the surface process. Thus,

$$\frac{d\nu}{dt}\propto\nabla[\mu(1-l)\nabla\nu]. \quad (29)$$

This nonlinear diffusion operation on ν reduces noises in the final line process result.

5 Experimental Results

5.1 *Feature Extraction*

For our experiment, Law's texture filters are used as a part of the texture feature extraction process. There are five 1-D filters producing possible 25 features (5 different filters in horizontal and vertical directions yields 25 different combination of filters) at every pixel location. Law's filters are

$$h_1=\{1\ \ 4\ \ 6\ \ 4\ \ 1\},$$

$$h_2=\{-1\ \ -2\ \ 0\ \ 2\ \ 1\},$$

$$h_3=\{-1\ \ 0\ \ 2\ \ 0\ \ -1\},$$

$$h_4=\{-1\ \ 2\ \ 0\ \ -2\ \ 1\},\quad \text{and}$$

$$h_5=\{1\ \ -4\ \ 6\ \ -4\ \ 1\}. \quad (30)$$

The next operation after Law's filter operation is a pixelwise nonlinear transformation using hyperbolic tangent. For each pixel in the filter output images, pixel values is modified by

$$y=\tanh(\beta x), \quad (31)$$

where β is a constant that is set to 1.0 throughout the experiments, x is the pretransformation value, and y is the new value. This nonlinear transform enhances the contrast of the features.

Following the nonlinear transform is the local average operation. The average absolute deviation described in Ref. 1 is used. A concentric Gaussian is used as a window function for the averaging. The output of the average operation is

$$z=G*J, \quad (32)$$

where

$$J(x;y;I)=|I(x,y)|, \quad (33)$$

$I(x,y)$ is the input image, and G is the concentric Gaussian. The size and the standard deviation of the Gaussian window must be adjusted to match the sizes of texels in each image. Figure 2 shows the overall texture feature extraction process.

Normalized pixel values are used as color features. For each input image, pixel values are shifted and scaled so that the mean of the pixel values are 0 and the variance is 1.0.

5.2 *Results*

Note that throughout the experiments, the color parameters $(\alpha_c,\mu_c,\nu_{0c})$ and the texture counterparts $(\alpha_t,\mu_t,\nu_{0t})$ are set to be equal for the blending through surface methods so that we introduce a minimum amount of parameter adjustment. On the other hand, we allowed two sets to be different for the blending through line processes.

Figures 3 and 4 show the results of two algorithms applied to synthetic images. The left images are the original images, the center images are the results of the blending through surface processes, and the right images are the re-

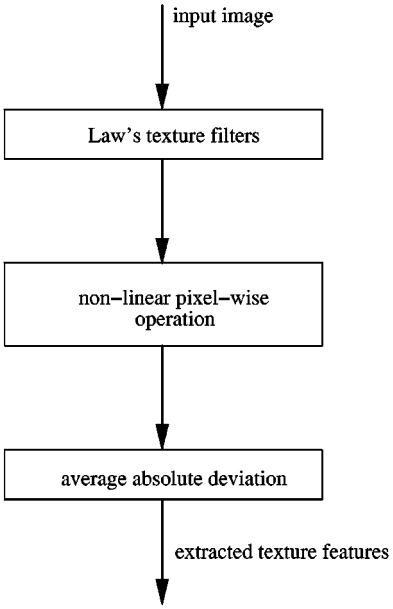


Fig. 2 Texture feature extraction process.

sults of the blending through line processes. The values of the parameters are given in the caption of each figure. The left image in Fig. 3 is the same test image shown in Fig. 1.

For Fig. 3, the blending through surface processes detected most of the boundaries accurately with few spurious boundaries inside the textured regions. The blending through line processes detected most of the boundaries, but they are thicker and less accurate than the blending through surface process. The results of the blending through surface processes shows the most complete, most accurate, and least noisy result among those shown in Fig. 1 and 3.

For Fig. 4, the blending through surface processes detected much more complete boundaries than the blending through line processes while picking up less boundaries.

Figures 5 and 6 show the results of boundary detection applied to natural images. The top-left images are the original images, the top-right images are the results using only the color feature, the bottom-left images are the results of the blending through surface processes, and the bottom-right images are the results of the blending through line processes. The values of the parameters are given in the caption of each figure. Figure 5 consists of two sofas, a plain one (the one closer to the viewer) and a colorful one. Figure 6 consists of two moving cars, a textured surface, and a nontextured but noisy surface.

For Fig. 5, the result with only the color feature could not distinguish the true region boundaries from the texture edges within the colorful sofa pattern. The blending through surface processes detected the boundaries of the plain sofa, although the boundary for the colorful sofa is noisy. The blending through line processes did not detect the sofa boundaries as well as the blending through surface processes, however, it has much less spurious boundaries than the result with only the color feature (top right).

For Fig. 6, all results look similar, but the bottom two (results through the blending) are less noisy and have better boundary representation.

Figure 7 shows the final parameter images for ν and λ when the blending through surface processes is applied to the sofa image (Fig. 5). Note that the parameters are smooth within the region boundaries, λ is small within heavily textured regions while ν is large in the region.

6 Conclusion

This paper introduced two algorithms for boundary detection using multiple feature sets, mainly color and texture features. It also described a heuristic parameter adaptation scheme to enhance the performance of the algorithms.

In all the experiments, the blending through surface processes outperformed the blending through line processes and no blending (only using the color features). It was rather surprising that without any prior information, the algorithm tended to distinguish textured regions and nontextured regions, as can be seen in Fig. 7, through minimization of the energy Eq. (15).

The amount of parameter adjustment was very small for the blending through surface processes. This was another advantage over the blending through line processes.

Fig. 3 Feature blending results I. The result of boundary detection on a synthetic image with textured regions and nontextured regions: left, the original gray-scale image; middle, the result of the blending through surface processes using both gray-scale pixel values and texture features ($\alpha=0.5$, $\mu=1.0$, and $\nu 0=0.02$); and right, the result of the blending through line processes ($\alpha_c=\alpha_t=0.1$, $\mu_c=\mu_t=1.0$, $\nu 0_c=0.03$, $\nu 0_t=0.5$, and $\lambda=4$).

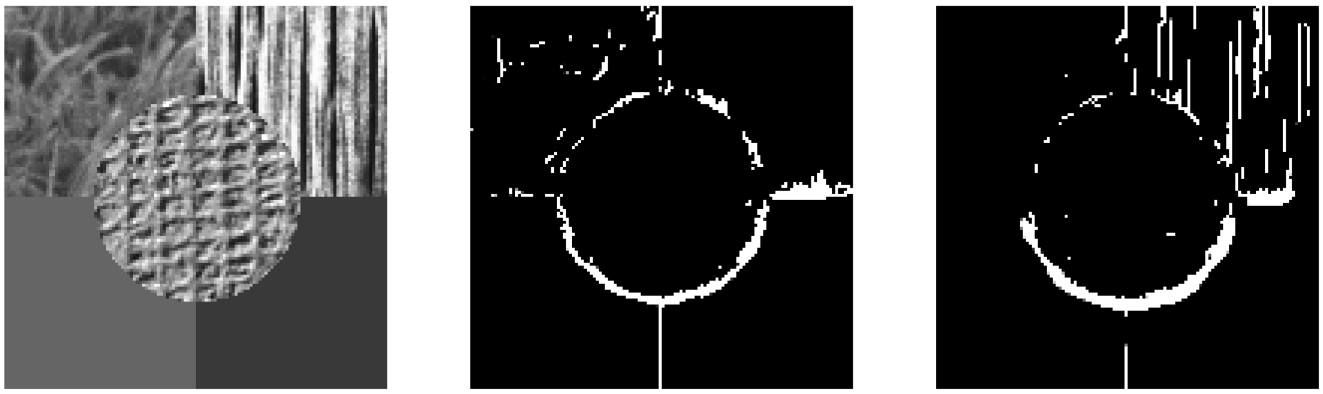


Fig. 4 Feature blending results II. The result of boundary detection on a synthetic image with textured regions and nontextured regions: left, the original gray-scale image; middle, the result of the blending through surface processes using both gray-scale pixel values and texture features ($\alpha=0.3$, $\mu=1.0$, and $\nu 0=0.03$); and right, the result of the blending through line processes ($\alpha_c=\alpha_t=0.1$, $\mu_c=\mu_t=1.0$, $\nu 0_c=0.01$, $\nu 0_t=0.5$, and $\lambda=4$).

Throughout the experiments, only two parameters were involved for adjustment (α and $\nu 0$).

To show the effectiveness of the feature blending, WMM was applied on the synthetic images (Figs. 1 and 4) using the color features and the texture features independently, thus generating two separate boundary results for each synthetic image. Given two boundary images, the optimal but artificial blending is performed by the following: the location is a part of a boundary if both results agree that it is on a boundary or if one of the results shows that it is on a boundary and the location is indeed on a true boundary.

The artificially (in a sense that the process requires the knowledge of the true boundaries) generated boundary results for two synthetic images are shown in Fig. 8. These results are more accurate than the others shown previously. However, the results of the blending through surface processes (the middle images of Figs. 3 and 4) are very comparable to the optimal results, thus the blending method produced near optimal results on these synthetic images.

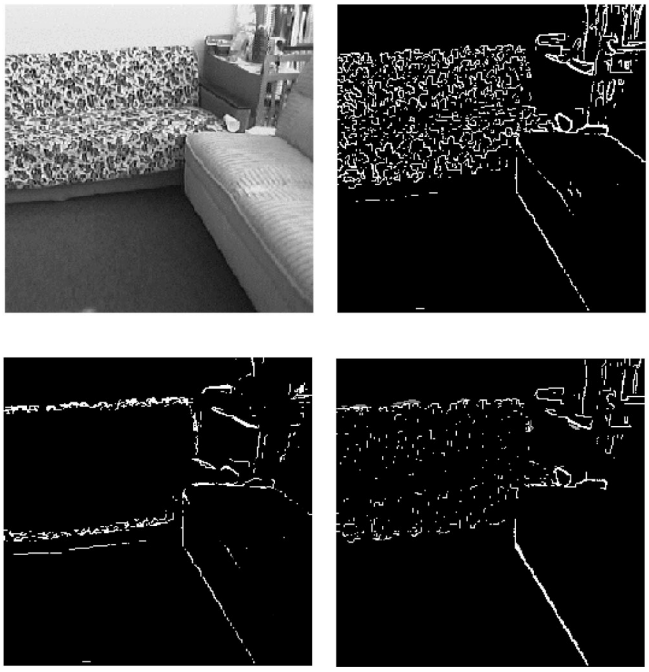


Fig. 5 Feature blending results III. The result of boundary detection on a room with sofa image: top left, the original gray-scale image; top right, the result of boundary detection using only gray-scale pixel values ($\alpha=0.3$, $\mu=1.0$, and $\nu 0=0.03$); bottom left, the result of the blending through surface processes using both gray-scale pixel values and texture features ($\alpha=0.3$, $\mu=1.0$, and $\nu 0=0.02$); and bottom right, the result of the blending through line processes ($\alpha_c=\alpha_t=0.5$, $\mu_c=1.0$, $\mu_t=2.0$, $\nu 0_c=\nu 0_t=0.1$, and $\lambda=8$).

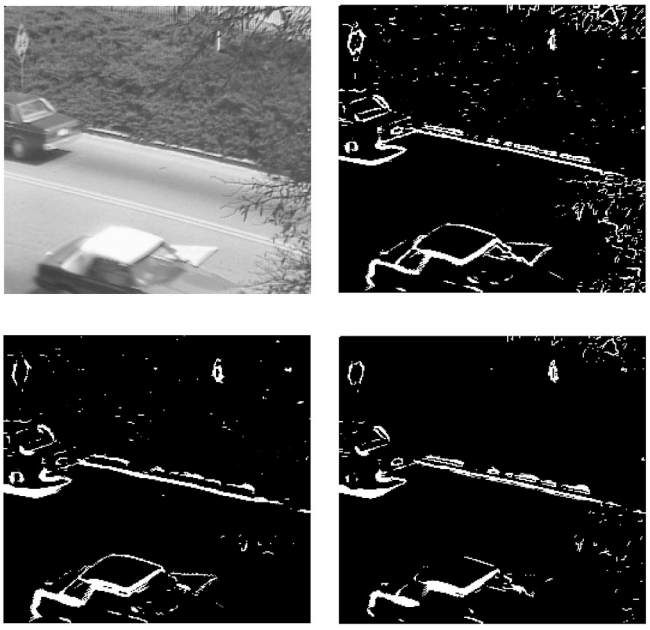


Fig. 6 Feature blending results IV. The result of boundary detection on a natural scene image: top left, the original gray-scale image; top right, the result of boundary detection using only gray-scale pixel values ($\alpha=0.3$, $\mu=1.0$, and $\nu 0=0.03$); bottom left, the result of the blending through surface processes using both gray-scale pixel values and texture features ($\alpha=0.3$, $\mu=1.0$, and $\nu 0=0.03$); and bottom right, the result of the blending through line processes ($\alpha_c=\alpha_t=0.4$, $\mu_c=1.0$, $\mu_t=4.0$, $\nu 0_c=\nu 0_t=0.1$, and $\lambda=16$).

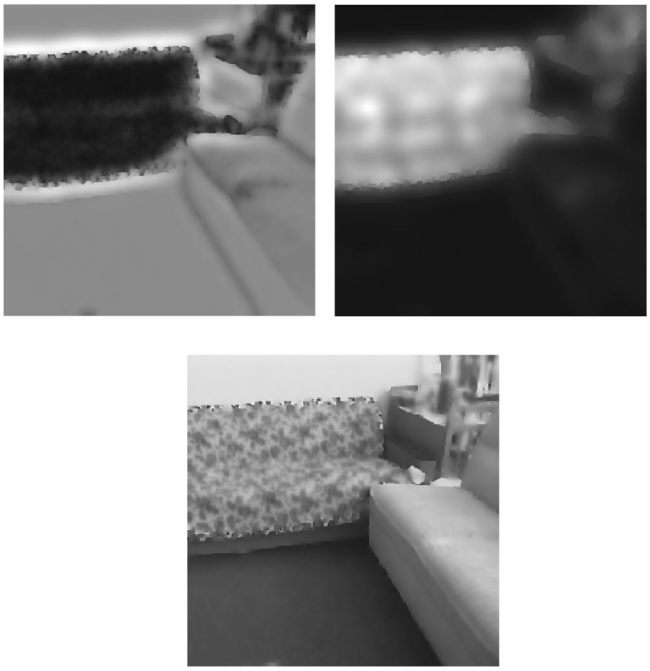


Fig. 7 Final diffused parameter image: top left, the final parameter λ image plane; top right, the final parameter $\nu 0$ image plane; and bottom, the final color feature image plane. Note the parameters are smooth within a region.

Due to the complexity of the systems [Eqs. (8) and (6)], it is difficult to analytically compare the performance of the two methods. In our opinion, the superior performance of the blending through surface process is attributed to the fact that for the blending through surface processes, the coupling is done through the parameter λ, which covers the whole 2-D area, while for the blending through line processes, the coupling is done through the line processes, which covers only a subset of the 2-D image area. Thus, the former has a better control of coupling over the whole image area. The time variable B in Eq. (20) was introduced to complement this insufficiency, and it could be possible to improve the performance by using a different scheduling from Eq. (23).

The success of treating λ as another surface process and letting it participate in the minimization process has directed our attention to the idea of treating all the parameters similarly. Thus, the parameters adapt themselves as the result of the energy minimization process by the parameters. New energy constraints must be devised and tested. This is our future direction for more robust scene segmentation/boundary detection mechanism.

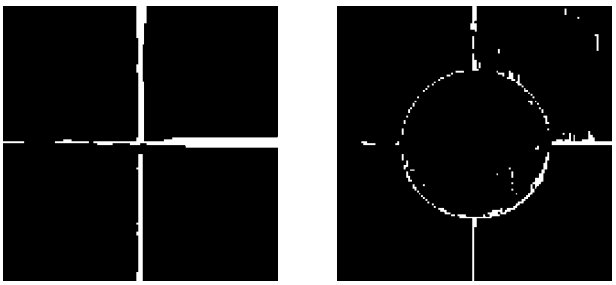


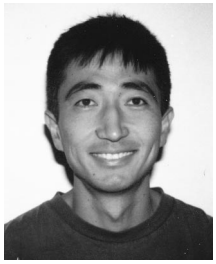
Fig. 8 Artificially generated optimal blending results: left, the result for the synthetic image in Fig. 1; right, the result for the synthetic image in Fig. 4. For more detail, see the text.

Acknowledgment

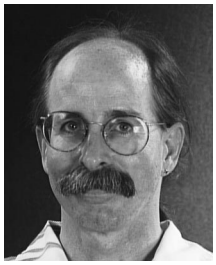
This research is supported in part under Office of Naval Research grant no. N00014-94-1-1163 and Army Research Office grant no. DAAH04-96-10326.

References

1. A. K. Jain and F. Farrokhnia, ''Unsupervised texture segmentation using Gabor filters,'' *Pattern Recog.* **24**(12), 1167–1186 (1991).
2. T. S. Lee, ''A Bayesian framework for understanding texture segmentation in the primary visual cortex,'' *Vis. Res.* **35**(18), 2643–2657 (1995).
3. D. Geiger and A. Yuille, ''A common framework for image segmentation,'' *Int. J. Comput. Vis.* **6**(3), 227–243 (1991).
4. A. K. Jain, ''Partial differential equations and finite difference methods in image processing—part I: image representation,'' *J. Optimiz. Theory Appl.* **23**, 65–91 (Sep. 1977).
5. A. K. Jain and J. R. Jain, ''Partial differential equations and finite difference methods in image processing—part II: image restoration,'' *IEEE Trans. Automat. Control* **23**(5), 596–613 (1978).
6. S. Geman and D. Geman, ''Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images,'' *IEEE Trans. Pattern Anal. Mach. Intell.* **6**(6), 721–741 (1984).
7. A. Blake and A. Zisserman, *Visual Reconstruction*, MIT Press, Cambridge, MA (1987).
8. D. Mumford and J. Shah, ''Boundary detection by minimizing functionals,'' in *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pp. 22–26 (1985).
9. J. J. Hopfield, ''Neurons with graded response have collective computational properties like those of two-state neurons,'' *Proc. Natl. Acad. Sci. U.S.A.* **81**, 3088–3092, 1984.
10. T. Kubota and T. Huntsberger, ''Adaptive anisotropic parameter estimation in the weak membrane model,'' in *Proc. Intl. Workshop on Energy Minimization Methods in Computer Vision and Pattern Recog.*, pp. 179–194, Venice, Italy (1997).
11. C. Marroquin, C. Koch, and A. Yuille, ''Analog 'neuronal' networks in early vision,'' *Proc. Natl. Acad. Sci. U.S.A.* **83**, 4263–4267 (1985).
12. D. Geiger and F. Girosi, ''Parallel and deterministic algorithms from MRF's: surface reconstruction,'' *IEEE Trans. Pattern Anal. Mach. Intell.* **13**(5), 401–412 (1991).



Toshiro Kubota received his BS degree in instrumentation engineering from Keio University in 1988 and his MSEE and PhD degrees in electrical and computer engineering from Georgia Institute of Technology in 1989 and 1995, respectively. From 1989 to 1995, he was a research assistant with the Computer Engineering Research Laboratory, Georgia Institute of Technology. In 1996, he became a lecturer/research associate with the Department of Computer Science in University of South Carolina. His current research interests include computer vision, biological vision, neural networks, parallel processing, and visualization.



Terry Huntsberger received his PhD in physics from the University of South Carolina in 1978, where he currently directs the Intelligent Systems Laboratory and is an associate professor in the Department of Computer Science. His current research interests include waveletbased image analysis, computer graphics, and parallel simulation of physical processes. He is a member of the SPIE, ACM, INNS, and the IEEE Computer Society.